

09/01/99



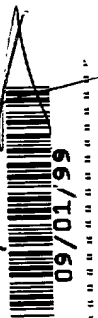
Jc674 U.S. PTO

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventorship..... Knight et al.
Applicant..... Microsoft Corporation
Attorney's Docket No. MS1-321US
Title: System and Method for Data Synchronization

Jc525 U.S. PTO

09/388829



09/01/99

TRANSMITTAL LETTER AND CERTIFICATE OF MAILING

To: Commissioner of Patents and Trademarks
Washington, D.C. 20231

From: Steven R. Sponseller (509) 324-9256
Lee & Hayes, PLLC
421 W. Riverside Avenue, Suite 500
Spokane, WA 99201

The following enumerated items accompany this transmittal letter and are being submitted for the matter identified in the above caption.

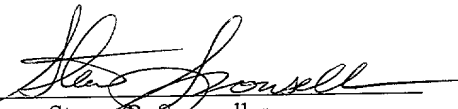
1. Transmittal Letter with Certificate of Mailing included.
2. PTO Return Postcard Receipt
3. New patent application (title page plus 30 pages, including claims 1-33 & Abstract)
4. Executed Declaration
5. 7 sheets of formal drawings (Figs. 1-7)
6. Assignment w/Recordation Cover Sheet

Large Entity Status ☒

Small Entity Status ☐

The Commissioner is hereby authorized to charge payment of fees or credit overpayments to Deposit Account No. 50-0463 in connection with any patent application filing fees under 37 CFR 1.16, and any processing fees under 37 CFR 1.17.

Date: 9-01-99

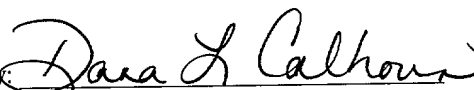
By: 
Steven R. Sponseller
Reg. No. 39,384

CERTIFICATE OF MAILING

I hereby certify that the items listed above as enclosed are being deposited with the U.S. Postal Service as either first class mail, or Express Mail if the blank for Express Mail No. is completed below, in an envelope addressed to The Commissioner of Patents and Trademarks, Washington, D.C. 20231, on the below-indicated date. Any Express Mail No. has also been marked on the listed items.

Express Mail No. (if applicable) EL 4 27262187

Date: 9/1/99

By: 
Dana L. Calhoun

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

System and Method for Data Synchronization

Inventor(s):

Kenneth J. Knight

David J. Messner

1 **TECHNICAL FIELD**

2 This invention relates to data synchronization. More particularly, the
3 invention relates to synchronizing multiple data caches contained in multiple
4 servers using a central database.
5

6 **BACKGROUND OF THE INVENTION**

7 A variety of internet-based server applications require access to certain
8 data. In situations where multiple servers are executing multiple instances of a
9 particular server application, all instances of the server application require access
10 to the same set of data. For example, a commerce-related server application
11 requires access to a set of data containing information regarding product (or
12 service) prices, shipping charges, and promotions or other discounts. If multiple
13 servers are executing the same commerce-related server application, each instance
14 of the application must access the same set of data to accurately and consistently
15 calculate the price of a customer's order. If different instances of the commerce-
16 related server application access different sets of data, different instances of the
17 application may calculate different prices for the same order. To avoid this
18 problem, it is important that all instances of a particular application access the
19 same set of data or access different sets of synchronized data.

20 In a typical collection of web servers, referred to as a "web farm", a
21 technique known as "DNS round-robin load balancing" is often used to distribute
22 tasks among the multiple web servers. Using this technique, each web server in
23 the web farm is assigned a unique internet protocol (IP) address. In this situation,
24 a single internet site address is associated with a list of IP addresses (i.e., the
25 unique IP addresses assigned to each web server in the web farm). When a client

1 browser resolves an internet site address using a domain name service (DNS)
2 lookup, the client browser receives the list of IP addresses associated with all web
3 servers in the web farm. In response to the first request, the client browser selects
4 one entry in the list as the starting point. The client browser then rotates through
5 the list of addresses in a round-robin manner for each subsequent request. When
6 the browser reaches the end of the list, the next request is retrieved from the
7 beginning of the list. Thus, each time a web page associated with an instance of
8 an internet-based application is accessed, the next web server in the web farm (i.e.,
9 the web server associated with the next IP address in the list of addresses) is used
10 to provide the web page to the browser.

11 In the internet server application discussed above, all web servers in the
12 web farm must access the same set of data regarding pricing, shipping, and
13 discounts. If different web servers apply different sets of data, then the price of
14 the customer's order may change with each new web page access. For example, a
15 simple "web page refresh" command will cause the browser to retrieve the
16 "refreshed" web page from the web server associated with the next IP address in
17 the list. If the new web server applies a different set of data to determine pricing,
18 shipping costs, and discounts, then the price displayed to the customer may change
19 after the page is refreshed, even though the actual order has not changed. This
20 situation is undesirable and may cause the customer to abandon the web site,
21 thereby resulting in lost revenue (and possibly a lost customer) for the operator of
22 the site.

23 To ensure that all web servers access the same set of data, some existing
24 systems use a two-tier approach of the type shown in Fig. 1, which allows multiple
25 web servers to access data stored in a database through a cache server. Fig. 1

1 shows a web farm 10 comprised of multiple web servers 12. Each web server 12
2 is connected to a cache server 14, which is connected to a database 16. Web
3 servers 12 are connected to cache server 14 through a local area network (LAN)
4 18. To access a particular set of data, one of the web servers 12 issues a data
5 request across LAN 18 to cache server 14. Cache server 14 then retrieves the
6 requested data from database 16 and provides the retrieved data across LAN 18 to
7 the requesting web server 12. By providing a common database, each web server
8 12 accesses the same set of data when generating a web page.

9 Although the system of Fig. 1 ensures that each web server accesses a
10 common set of data, the use of a single cache server 14 introduces a single point of
11 failure. If cache server 14 fails, then all web servers 12 are prevented from
12 accessing data contained in database 16. Furthermore, the system shown in Fig. 1
13 creates a significant amount of network traffic on LAN 18. Each time a web
14 server 12 requests data from database 16, several messages (as well as the
15 requested data) are sent across LAN 18 to satisfy the request. If the data requested
16 by the web servers 12 does not change frequently, the web servers 12 may request
17 the same data numerous times before the data in database 16 changes. This
18 repeated transmission of data requests for the same data generates significant
19 network traffic that is unnecessary if the data has not changed since the last data
20 request. Additionally, if network traffic is heavy, the time required to retrieve data
21 across LAN 18 may significantly delay the generation of a web page by web
22 server 12. If data retrieval speed is important, then the delays associated with the
23 system of Fig. 1 may prevent acceptable operation of web server 12.

1 **SUMMARY OF THE INVENTION**

2 The invention allows multiple web servers to cache data locally, while still
3 maintaining data synchronization among themselves. This is accomplished
4 without requiring any web server to have knowledge of the other web servers.
5 Instead, each web server communicates with a common data server to retrieve data
6 from a common database. In addition to storing data, the data server indicates a
7 time at which the data should become active within each web server.

8 The data retrieved from the data server is initially stored by each web server
9 in a local staging cache until reaching the synchronization time indicated by the
10 data server. When the synchronization time arrives, all web servers copy the data
11 from their staging cache to the active cache at approximately the same time. Thus,
12 all web servers maintain the same set of active data although the web servers do
13 not communicate with one another and are unaware of the other web servers.

14 An implementation of the invention synchronizes data among multiple web
15 servers, each coupled to a common data server, by retrieving a scheduled
16 activation time from the data server. If the current time is prior to the scheduled
17 activation time, then each web server retrieves updated data into a staging cache in
18 the web server. At the scheduled activation time, each web server copies data
19 from its staging cache to an active cache in the web server.

20 Other aspects of the invention provide that after the scheduled activation
21 time, the first web server to initiate a retrieval process updates data caches in the
22 data server and calculates a next scheduled activation time.

23 In the described implementation of the invention, clock synchronization
24 issues between the web servers and the data server are addressed by maintaining a
25 time difference between the clock in the data server and the clock in each web

1 server. This time difference is taken into account when each web server
2 determines its next scheduled synchronization time.

3 In accordance with another aspect of the invention, the retrieval of updated
4 data into staging caches in the plurality of web servers is performed
5 asynchronously.

6 When a new web server is added to the multiple web servers, data is copied
7 from an active cache in the data server to an active cache in the new web server.
8

9 **BRIEF DESCRIPTION OF THE DRAWINGS**

10 Fig. 1 illustrates a prior art system that allows multiple web servers to
11 access data stored in a database through a cache server.

12 Fig. 2 illustrates an exemplary system that allows multiple web servers
13 coupled to a data server to retrieve data from the data server.

14 Fig. 3 is a block diagram showing pertinent components of a computer in
15 accordance with the invention.

16 Fig. 4 is a block diagram illustrating pertinent components of a web server
17 and a data server in accordance with the invention.

18 Fig. 5 is a flow diagram illustrating an exemplary procedure performed by
19 each web server during the retrieval phase of each data synchronization cycle.

20 Fig. 6 is a flow diagram illustrating an exemplary procedure performed by
21 each web server to identify the next activation time and implement the activation
22 phase of the data synchronization cycle.

23 Fig. 7 is a time line illustrating an exemplary series of events for
24 synchronizing data in multiple web servers using a common data server.
25

DETAILED DESCRIPTION

Fig. 2 illustrates an exemplary system that allows multiple web servers coupled to a data server to retrieve data from the data server. The system includes a web farm 100, which comprises multiple web servers 102. The web farm 100 is accessed by one or more client computers, such as computers using internet browsers to access the web farm through the Internet. Multiple client computers can access the web farm 100 simultaneously, thereby allowing the users of the client computers to retrieve web pages from the web servers 102 simultaneously. For example, if each web server 102 is executing an internet-based electronic commerce application, then the users of the client computers can simultaneously access the web servers 102 in web farm 100 to shop for and purchase goods or services offered by the operator of the web site.

Each web server 102 maintains and accesses a set of cached data 104 when generating web pages and performing other tasks. By storing data locally in the web servers 102, each web server can quickly retrieve the data without waiting for network transfer times and other delays associated with retrieving data from a remote device. If numerous web servers 102 are repeatedly accessing data across the network, significant delays may result in the generation of web pages by the web servers. These delays are significantly reduced by the local caching of data within the web servers 102, thereby resulting in faster web page generation.

The web servers 102 are arranged and accessed using the DNS round-robin technique discussed above. Therefore, it is important that each web server 102 contain the same set of data 104 when generating web pages. Otherwise, the web pages may differ from one web server to the next (e.g., displaying different pricing

1 information) even though the data (e.g., price and discount percentage) should not
2 change from one web server to the next. As discussed above, if different web
3 servers 102 access different data when generating web pages, a simple web page
4 refresh operation may change the information displayed on the web page because
5 a different web server generates the subsequent web page using different data. To
6 avoid this situation, the present invention provides a mechanism for
7 simultaneously synchronizing the various sets of data 104 among all web servers
8 102.

9 The web servers 102 are coupled to a data server 106 via communication
10 links 108. In most cases, the communications links will be formed by local area
11 network connections, although other types of communications might be utilized
12 such as wide area network connections and dedicated computer-to-computer
13 connections.

14 Data server 106 contains data and other information used by web servers
15 102 to generate web pages and to synchronize data at the appropriate time. Data
16 server 106 thus acts as the central storage location for data that is to be retrieved
17 by each web server 102. In one embodiment of the invention, data is stored in
18 data server 106 using a SQL table or SQL database.

19 A set of management tools 110 are used to configure and manage the
20 operation of data server 106 (e.g., manage the data that is copied from the data
21 server to the web servers). Additionally, management tools 110 are able to set and
22 modify various data values and other information stored within data server 106.
23 The management tools also allow modification of the time between
24 synchronization cycles. In the described embodiment, management tools 110 are
25 programs that are executed on data server 106, although they could alternatively

1 be executed from one of the web servers 102 or from another computer that has
2 communications with data server 106.

3 Fig. 3 shows a general example of a desktop computer 130 that can be used
4 in accordance with the invention. A computer such as that shown in Fig. 3 can be
5 used for any of the web servers 102 or data server 106.

6 Computer 130 includes one or more processors or processing units 132, a
7 system memory 134, and a bus 136 that couples various system components
8 including the system memory 134 to processors 132. The bus 136 represents one
9 or more of any of several types of bus structures, including a memory bus or
10 memory controller, a peripheral bus, an accelerated graphics port, and a processor
11 or local bus using any of a variety of bus architectures. The system memory 134
12 includes read only memory (ROM) 138 and random access memory (RAM) 140.
13 A basic input/output system (BIOS) 142, containing the basic routines that help to
14 transfer information between elements within computer 130, such as during start-
15 up, is stored in ROM 138.

16 Computer 130 further includes a hard disk drive 144 for reading from and
17 writing to a hard disk (not shown), a magnetic disk drive 146 for reading from and
18 writing to a removable magnetic disk 148, and an optical disk drive 150 for
19 reading from or writing to a removable optical disk 152 such as a CD ROM or
20 other optical media. The hard disk drive 144, magnetic disk drive 146, and optical
21 disk drive 150 are connected to the bus 136 by an SCSI interface 154 or some
22 other appropriate interface. The drives and their associated computer-readable
23 media provide nonvolatile storage of computer-readable instructions, data
24 structures, program modules and other data for computer 130. Although the
25 exemplary environment described herein employs a hard disk, a removable

1 For purposes of illustration, programs and other executable program
2 components such as the operating system are illustrated herein as discrete blocks,
3 although it is recognized that such programs and components reside at various
4 times in different storage components of the computer, and are executed by the
5 data processor(s) of the computer.

6 Fig. 4 is a block diagram illustrating pertinent components of one of the
7 web servers 102 and the data server 106 in accordance with the invention. The
8 web server 102 includes a staged data cache 200 and an active data cache 202.
9 Active data cache 202 contains current data that web server 102 is currently using
10 to generate web pages and perform other tasks. The active data caches 202 of all
11 web servers 102 in the web server farm contain the same set of data, so that all
12 web servers use the same data to generate the same web page. The staged data
13 cache 200 contains data that will eventually become the current data, either by
14 being copied into the active data cache 202 or by designating the staged data cache
15 as the active data cache. As discussed in greater detail below, each web server 102
16 in the web farm copies the data from its staged data cache 200 to its active data
17 cache 202 at the same time, such that the active data in all web servers 102
18 remains synchronized.

19 Data server 106 includes a production data cache 204, a staged data cache
20 206, and an active data cache 208. The production data cache 204 stores data
21 being created or modified using one or more management tools 110. The data
22 stored in cache 204 may not be complete and is not necessarily ready for use in
23 generating web pages. Production data cache 204 may also store data that is
24 scheduled to be activated at a particular date and time. This scheduled data is
25 stored in the production data cache 204 until the scheduled time has occurred or

1 will occur in the near future (e.g., within the next ten minutes), at which time the
2 data is copied to the staged data cache 206, as discussed below.

3 The staged data cache 206 contains a copy of updated data after all
4 modifications have been performed in production data cache 204. This is data that
5 will eventually become the current data, for consumption by the web servers 102.
6 The active data cache 208 contains the actual data that is provided to individual
7 web servers upon request. As will be described in more detail below, the web
8 servers request the data in prearranged synchronization cycles. Whenever the
9 staged data cache of the data server contains updated data, this data is copied to
10 the staged data cache 200 of the web server during the first phase of each
11 synchronization cycle. All web servers copy data from their staged data cache 200
12 to their active data cache 202 at the next scheduled activation time, such that all
13 web servers begin using the same updated data at approximately the same time.

14 Data server 106 also includes a synchronization table 210. The
15 synchronization table 210 includes information relating to the date and time of an
16 upcoming data synchronization cycle. Synchronization table 210 also includes
17 information regarding the desired time intervals between data synchronization
18 cycles. For example, the table might indicate that web servers 102 are to perform
19 a data synchronization cycle every ten minutes.

20 Each synchronization cycle includes two phases: a retrieval phase and an
21 activation phase. During the retrieval phase, each web server 102 queries the data
22 server 106 for a fresh copy of data from the staged data cache 206 of the data
23 server. This data is written to the staged data cache 200 of the individual web
24 servers 102. Because the retrieved data does not immediately become active (it is
25 not immediately put into the active data cache of the web server), it is not

1 necessary for the retrieval operations to be closely synchronized between the
2 individual web servers. Thus, the retrieval phase is an asynchronous process.

3 The activation phase comprises "activating" the data that was previously
4 retrieved from staged data cache 206 of the data server. This involves, for each
5 web server 102, either copying the contents of staged data cache 200 to active data
6 cache 202, or switching the designations of the two caches so that the staged data
7 cache now becomes the active data cache. This activation step is performed by all
8 web servers at a single prearranged time. Thus, the activation phase is a
9 synchronous process. The retrieval phase is performed at defined intervals.

10 Fig. 5 is a flow diagram illustrating an exemplary procedure performed by
11 each web server 102 during the retrieval phase of each data synchronization cycle.
12 The retrieval phase represents the start of a new synchronization cycle and follows
13 the activation phase of the previous synchronization cycle. The procedure of Fig.
14 5 is triggered when the previous activation phase is complete (i.e., the activation
15 time has passed).

16 At step 220, the web server executing the procedure locks the
17 synchronization table 210, thereby preventing other web servers from accessing
18 the data server while the current web server is retrieving, and possibly modifying,
19 data stored in the data server. This locking of the data server prevents two web
20 servers from attempting to simultaneously modify the same data in the data server.
21 Additionally, locking the data server prevents more than one web server from
22 initiating the next synchronization cycle and ensures a consistent view of the data
23 for all web servers.

24 The web server continues the procedure by retrieving the synchronization
25 time from the synchronization table and retrieving the current system time from

1 the data server (step 222). At step 224, the web server compares the current time
2 to the synchronization time. If the current time is greater than the synchronization
3 time (i.e., the synchronization time has passed), then the procedure branches to
4 step 226, which marks the beginning of a new synchronization cycle.

5 At step 226, the web server calculates a new activation time by adding a
6 predefined synchronization interval to the previous activation time. The
7 synchronization interval is retrieved by the web server from the synchronization
8 table 210 in the data server. An example equation for this calculation:

$$\text{activation_time} = \text{activation_time} + \text{synchronization_interval}$$

11
12 The new activation time is stored in the synchronization table 210 in the data
13 server. The newly calculated activation time corresponds to the time of the next
14 activation phase - the time at which each of the web servers 102 will next copy
15 data from the web server's staged data cache 200 to the web server's active data
16 cache 202. Since each of the web servers will retrieve this new activation time
17 from the synchronization table, each of the web servers will update their active
18 data cache at approximately the same time.

19 Step 228 comprises updating the data server's active data cache 208 by
20 copying data from its staged data cache 206 to the active data cache. Step 230
21 comprises updating the data server's staged data cache 206 by copying data from
22 its production data cache 204 to the staged data cache. Thus, steps 226-230 set the
23 new (i.e., next) activation time and update the caches in the data server 106. The
24 procedure continues to step 232 to unlock the synchronization table, thereby
25

1 allowing other web servers to access the synchronization table, which now
2 contains the new activation time.

3 Since the new activation time (established in step 226) has not yet occurred,
4 all subsequent web servers that perform the procedure of Fig. 5 will continue from
5 step 224 to step 232 to unlock the synchronization table.

6 Thus, it is only necessary for one web server to update the activation time
7 (step 226) and update the data server caches (steps 228 and 230) during each
8 synchronization cycle. These updates are performed by the first web server to
9 execute the procedure of Fig. 5 after the previous synchronization cycle has
10 finished. After the first web server has performed steps 226-230, each subsequent
11 web server that executes the procedure of Fig. 5 during the same synchronization
12 cycle only needs to retrieve the updated data from the data server (step 234).

13 At step 234, each web server (including the web server that performed steps
14 226-230) updates its staged data cache 200 by copying data from the data server's
15 staged data cache 206 to the web server's staged data cache. At step 236, the web
16 server executing the procedure sets an internal timer or similar mechanism to
17 trigger at the new activation time. Each web server repeatedly attempts to perform
18 the procedure of Fig. 5 until it is successful (i.e., until the web server has copied
19 the appropriate data into its staged data cache 200 and retrieved the new activation
20 time).

21 The web servers have a period of time equal to the synchronization interval
22 value to copy data from the data server's staged data cache 206 to the web server's
23 staged data cache 200. To ensure that all web servers will have time to update
24 their staged data, the synchronization interval should be greater than the worst-
25

case time to copy the data from data server 106 to each web server 102. In an exemplary embodiment, the synchronization interval is ten minutes.

The clocks in the various web servers 102 and the data server 106 are not required to be synchronized. Instead, each web server compares its clock value to the clock value of the data server and maintains a time difference between the clock in the data server and the clock in the web server. This time difference is taken into account when each web server determines its next scheduled synchronization time. The clocks in web servers 102 may have different values based on different time zones or differences resulting from using different time sources to set the clocks on the web servers.

To ensure that all web servers 102 copy data simultaneously, each web server periodically identifies the time contained in the data server's clock. The web server then calculates the time difference between the data server's clock and the web server's clock. This difference is recorded by the web server and is used in all subsequent clock comparisons or calculations, such as comparing the web server's clock to the activation time.

The second phase of the synchronization cycle is referred to as the activation phase, which begins at the activation time. After the retrieval phase has finished, the procedure illustrated in Fig. 6 is initiated. Fig. 6 illustrates an exemplary procedure performed by each web server to identify the next activation time and implement the activation phase of the data synchronization cycle. Step 250 comprises the web server setting a timer event to occur at a particular time, defined by:

1 timer event = dt_Activation_time + dt_WebServer_time -
2 dt_DataServer_time

3
4 Where dt_Activation_time is the date and time of the next scheduled activation,
5 dt_WebServer_time is the date and time according to the clock in the web server,
6 and dt_DataServer_time is the date and time according to the clock in the data
7 server. Subtracting the data server time from the web server time determines the
8 difference between the two clocks. This difference is then applied to (i.e., added
9 to) the next scheduled activation time. Each web server calculates the timer event
10 in the same manner, thereby causing the timer event to occur at approximately the
11 same instant in time on all web servers.

12 Once the activation time has arrived (i.e., the timer event occurs), each web
13 server copies data from its staged data cache to its active data cache (step 252),
14 thereby overwriting the data previously stored in the active data cache. The
15 updating of the active data cache is the last process performed in the data
16 synchronization cycle. Once the active data cache update is complete, a new data
17 synchronization cycle is started.

18 When the activation phase is complete (i.e., all web servers have copied
19 data from their staged data cache to their active data cache), the synchronization
20 cycle is complete. The next synchronization cycle begins when the first web
21 server initiates the retrieval phase, as discussed above. The synchronization cycles
22 continue by alternating between retrieval phases and activation phases.

23 In the embodiment discussed above, data is copied from the web server's
24 staged data cache to its active data cache when the activation time arrives. In
25 another embodiment of the invention, instead of copying data from the staged data

If a new web server is coupled to the data server, then the active data cache 202 in the web server is initialized. This initialization is performed by copying data from the active data cache 208 in the data server 106 into the active data cache 202 in the new web server. This allows the new web server to begin operation immediately using current data rather than waiting for data to be copied into the web server's active data cache 202 during the next data synchronization cycle. A similar procedure is used if an existing web server is reset or was temporarily disconnected from the data server.

Thus, each web server 102 is responsible for copying data from its own staging cache to its active cache at the appropriate time. A separate cache server or similar device is not required to coordinate the synchronization of data among the multiple web servers 102. The configuration of the present invention reduces network communication traffic because each web server maintains a copy of the current data, which may be used to generate multiple web pages before the data is updated. Thus, rather than downloading the data each time the data is required by the web server, the data is retrieved by the web server once and stored locally until the next data retrieval phase. This local storage of data reduces the time required by the web server to generate web pages.

A particular embodiment of the invention has been described and illustrated herein with reference to multiple web servers coupled to a common data server.

1 However, the teachings of the present invention can be applied to any type of
2 server or other computing device that requires periodic updates of data from
3 another device, such as a data storage device.

4 Fig. 7 is a time line illustrating an exemplary series of events for
5 synchronizing data in multiple web servers using a common data server. Fig. 7
6 shows one complete synchronization cycle, starting on the left side of the time line
7 and ending on the right side of the time line. The synchronization cycle starts with
8 the retrieval phase and ends with the activation phase. The retrieval phase begins
9 when the activation phase of the previous synchronization cycle terminates. In the
10 retrieval phase, the first web server to perform the data retrieval process calculates
11 the next activation time and updates the data server's data caches (i.e., copies data
12 from the data server's staged data cache to the active data cache and from the
13 production data cache to the staged data cache). Next, each of the remaining web
14 servers identifies the next activation time (stored in the data server) and copies
15 data from the data server's staged data cache into the web server's staged data
16 cache. This retrieval of information by the web servers continues until all web
17 servers have retrieved the updated data from the data server.

18 When all web servers have retrieved the updated data, no further data
19 transfers occur until the activation phase begins, as determined by the activation
20 time. Since each web server has retrieved the same activation time and has
21 determined the time difference between its own clock and the clock of the data
22 server, all web servers copy data from their staged data cache to their active cache
23 at approximately the same time. After all web servers have copied data from their
24 staged data cache to their active cache, the synchronization cycle is complete. The
25

1 next synchronization cycle begins when the first web server initiates a data
2 retrieval process.

3 In the particular example of Fig. 7, one synchronization cycle takes
4 approximately ten minutes. The retrieval phase requires the first nine-and-a-half
5 minutes of the synchronization cycle and the activation phase requires the
6 remaining thirty seconds of the synchronization cycle. Other implementations of
7 the invention may utilize synchronization cycle times greater than ten minutes or
8 less than ten minutes depending on the amount of data retrieved by each web
9 server, the number of web servers coupled to the data server, and the frequency
10 with which the data in the data server is updated. As the amount of data retrieved
11 by each web server increases, the time necessary to complete the retrieval phase
12 also increases. As the number of web servers increases, the time required to
13 perform the retrieval phase also increases. If data in the data server is updated
14 infrequently, then the synchronization cycle time can be increased such that
15 synchronizations are performed less frequently. The length of the synchronization
16 cycle can be adjusted by changing the value of the synchronization interval,
17 discussed above.

18 In the embodiments described above, the data in the staged data cache is
19 always copied to the active data cache, regardless of whether the staged data is
20 different from the active data. In an alternate embodiment, the staged data is
21 copied to the active data cache only if the data has changed (i.e., the staged data is
22 different from the active data). This alternate embodiment can be applied in both
23 the web server and the data server. Similarly, the production data in the data
24 server is copied to the staged data cache in the data server only if the data has
25 changed. Further, the staged data in the data server is copied to the staged data

1 cache in the web server only if the data has changed. This alternate embodiment
2 can be implemented, for example, by using a flag or other indicator that is set or
3 changed when data in a data cache is modified.

4 Thus, a system has been described that allows multiple web servers to
5 maintain synchronized data without requiring the web servers to have knowledge
6 of any other web servers. Each web server executes a common procedure that
7 communicates with a data server to periodically retrieve updated data that is
8 copied to a staging area in the web server. At a particular time, all web servers
9 copy the staged data to an active data cache for use with all subsequent
10 calculations and web pages generated by the web server. The caching of data
11 locally on each web server allows fast retrieval of data from the cache, which
12 increases the speed at which web pages are generated and displayed to a user.

13 Although the invention has been described in language specific to structural
14 features and/or methodological steps, it is to be understood that the invention
15 defined in the appended claims is not necessarily limited to the specific features or
16 steps described. Rather, the specific features and steps are disclosed as preferred
17 forms of implementing the claimed invention.

1 **CLAIMS**

2 **1.** A method of synchronizing data among a plurality of web servers,
3 wherein each of the plurality of web servers is coupled to a common data server,
4 the method comprising:

5 retrieving a scheduled activation time from the data server;
6 prior to the scheduled activation time, retrieving updated data into staging
7 caches in the plurality of web servers; and
8 at the scheduled activation time, copying data from the staging cache of
9 each web server to an active cache of the web server.

10
11 **2.** A method as recited in claim 1 further comprising:
12 comparing a time associated with a clock in each web server to a time
13 associated with a clock in the data server; and
14 adjusting the scheduled activation time on each web server by the time
15 difference between the clock in the web server and the clock in the data server.

16
17 **3.** A method as recited in claim 1 wherein each web server contains a
18 clock, and wherein the clocks in the plurality of web servers are not synchronized
19 with one another.

20
21 **4.** A method as recited in claim 1 wherein the copying data comprises
22 swapping an active data cache pointer with a staged data cache pointer.
23
24
25

1 5. A method as recited in claim 1 wherein no communications are
2 required between the individual web servers to synchronize their data.

3
4 6. A method as recited in claim 1 wherein retrieving updated data into
5 staging caches in the plurality of web servers is performed asynchronously.

6
7 7. A method as recited in claim 1 further comprising:
8 after the scheduled activation time, updating data caches in the data server.

9
10 8. A method as recited in claim 1 further comprising:
11 after the scheduled activation time, calculating a next scheduled activation
12 time.

13
14 9. A method as recited in claim 1 further comprising:
15 after the scheduled activation time, updating data caches in the data server
16 and calculating a next scheduled activation time, wherein the updating and
17 calculating are performed by the first web server to initiate a retrieval process after
18 the scheduled activation time.

1 **10.** A method as recited in claim 1 further comprising:
2 if an additional web server is coupled to the data server, then copying data
3 from an active cache in the data server to an active cache in the additional web
4 server.

5
6 **11.** A method as recited in claim 1 further comprising:
7 if one of the plurality of web servers is initialized, then copying data from
8 an active cache in the data server to the active cache in the initialized web server.

9
10 **12.** A method as recited in claim 1 wherein the plurality of web servers
11 comprise a web farm.

12
13 **13.** A method as recited in claim 1 wherein the plurality of web servers
14 comprise a web farm, and wherein the plurality of web servers are load balanced
15 using a domain name service (DNS) round-robin technique.

16
17 **14.** One or more computer-readable memories containing a computer
18 program that is executable by a processor to perform the method recited in claim
19 1.

20
21 **15.** A system comprising:
22 a plurality of web servers coupled to a common data server, wherein each
23 of the plurality of web servers comprises:
24 a staging cache;
25 an active data cache coupled to the staging cache;

1 wherein the web server is configured to retrieve a scheduled
2 activation time from the data server, and further configured to retrieve
3 updated data from the data server into the staging cache prior to the
4 scheduled activation time; and

5 wherein the web server is configured to copy data from the staging cache to
6 the active data cache at the scheduled activation time.

7
8 **16.** A system as recited in claim 15 wherein each web server contains a
9 clock having an associated time, and wherein each web server is configured to
10 compare the time associated with the clock in the web server to a time associated
11 with a clock in the data server.

12
13 **17.** A system as recited in claim 16 wherein each web server is further
14 configured to adjust the scheduled activation time on the web server by the time
15 difference between the clock in the web server and the clock in the data server.

16
17 **18.** A system as recited in claim 15 wherein each web server contains a
18 clock, and wherein the clocks in the plurality of web servers are not synchronized
19 with one another.

20
21 **19.** A system as recited in claim 15 wherein the web server is further
22 configured to swap an active data cache pointer with a staged data cache pointer.

1 **20.** A system as recited in claim 15 wherein each of the plurality of web
2 servers is configured to update data caches in the data server after the scheduled
3 activation time.

4
5 **21.** A system as recited in claim 15 wherein each of the plurality of web
6 servers is configured to calculate a next scheduled activation time after the
7 scheduled activation time.

8
9 **22.** A system as recited in claim 15 wherein the plurality of web servers
10 comprise a web farm.

11
12 **23.** One or more computer-readable media having stored thereon a
13 computer program comprising the following steps:

14 retrieving a scheduled activation time from a data server;
15 prior to the scheduled activation time, retrieving updated data into a staging
16 cache in a server;
17 at the scheduled activation time, copying data from the staging cache in the
18 server to an active cache in the server; and
19 after the scheduled activation time, updating data caches in the data server
20 and calculating a next scheduled activation time.

1 **24.** One or more computer-readable media as recited in claim 23 further
2 comprising:

3 comparing a time associated with a clock in each server to a time associated
4 with a clock in the data server; and

5 adjusting the scheduled activation time on each server by the time
6 difference between the clock in the server and the clock in the data server.

7
8 **25.** One or more computer-readable media as recited in claim 23
9 wherein each server contains a clock, and wherein the clocks in the plurality of
10 servers are not synchronized with one another.

11
12 **26.** One or more computer-readable media as recited in claim 23
13 wherein updating data caches in the data server and calculating the next scheduled
14 activation time are performed if another process has not yet updated the data
15 caches or calculated the next scheduled activation time during a current data
16 synchronization cycle.

17
18 **27.** One or more computer-readable media as recited in claim 23 further
19 comprising:

20 if the server is initialized, then copying data from an active cache in the
21 data server to the active cache in the initialized server.

1 **28.** One or more computer-readable media as recited in claim 23
2 wherein the copying data comprises swapping an active data cache pointer with a
3 staged data cache pointer.
4

5 **29.** A method of synchronizing data among a plurality of web servers,
6 wherein each of the plurality of web servers is coupled to a common data server,
7 the method comprising:

8 providing a scheduled activation time from the data server to each of the
9 plurality of web servers;

10 communicating updated data into a staging cache in each of the plurality of
11 web servers prior to the scheduled activation time; and

12 copying data from the staging cache of each web server to an active cache
13 of the web server at the scheduled activation time.
14

15 **30.** A method as recited in claim 29 wherein the communicating
16 updated data into a staging cache is performed asynchronously.
17

18 **31.** A method as recited in claim 29 wherein the copying data comprises
19 swapping an active data cache pointer with a staged data cache pointer.
20

21 **32.** A method as recited in claim 29 wherein no communication is
22 required between the web servers to synchronize their data.
23
24
25

1 **33.** One or more computer-readable memories containing a computer
2 program that is executable by a processor to perform the method recited in claim
3 29.

667000 663330

1 **ABSTRACT**

2 Data is synchronized among multiple web servers, each of which is coupled
3 to a common data server. Each web server retrieves a scheduled activation time
4 from the data server. If the current time is prior to the scheduled activation time,
5 then each web server retrieves updated data from the data server into a staging
6 cache in the web server. At the scheduled activation time, each web server copies
7 data from its staging cache to an active cache in the web server. If a new web
8 server is added or an existing web server is initialized, then data is copied from an
9 active cache in the data server to an active cache in the new or initialized web
10 server. The multiple web servers may be arranged to form a web farm.
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

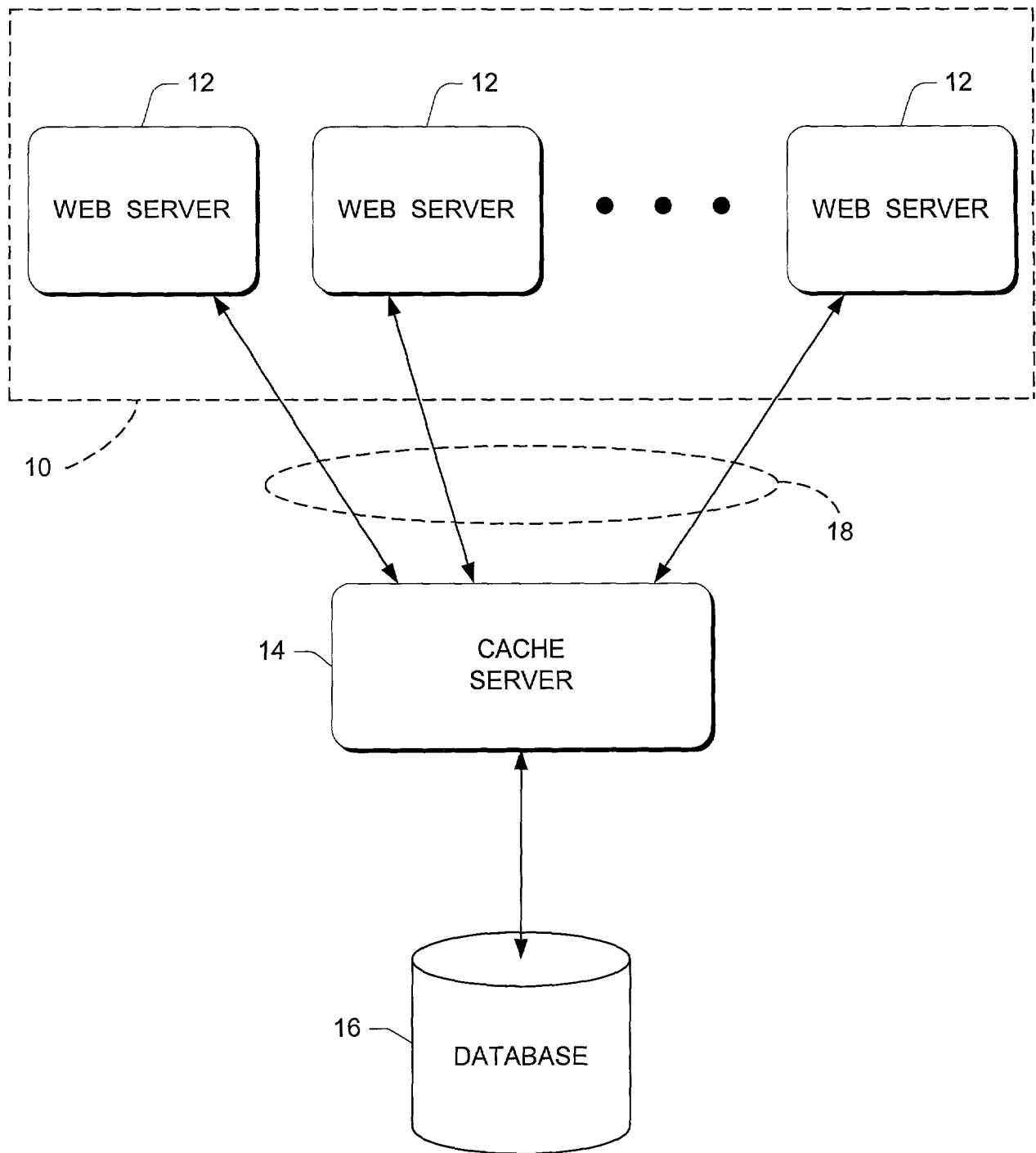


Fig. 1
(Prior Art)

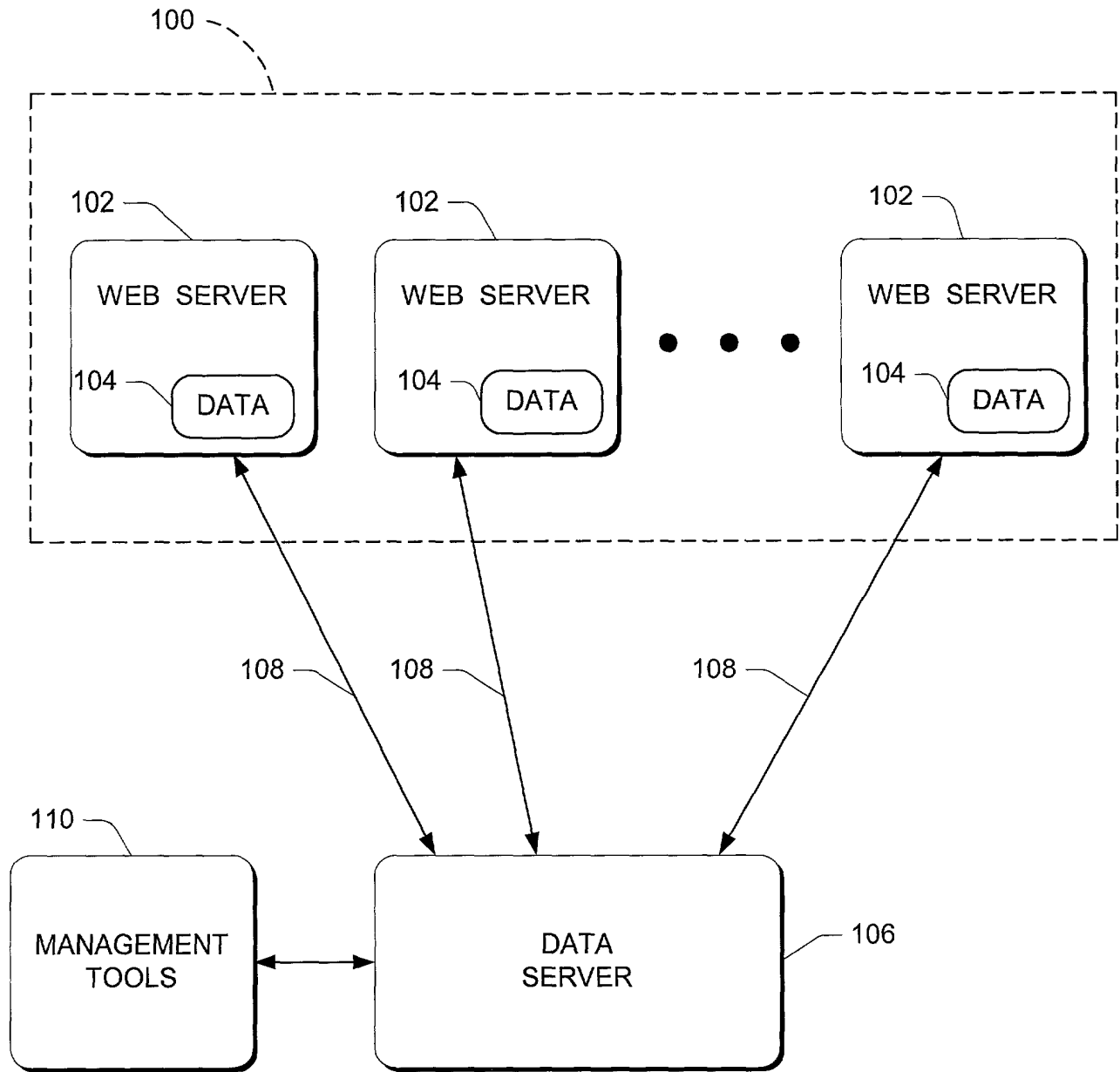
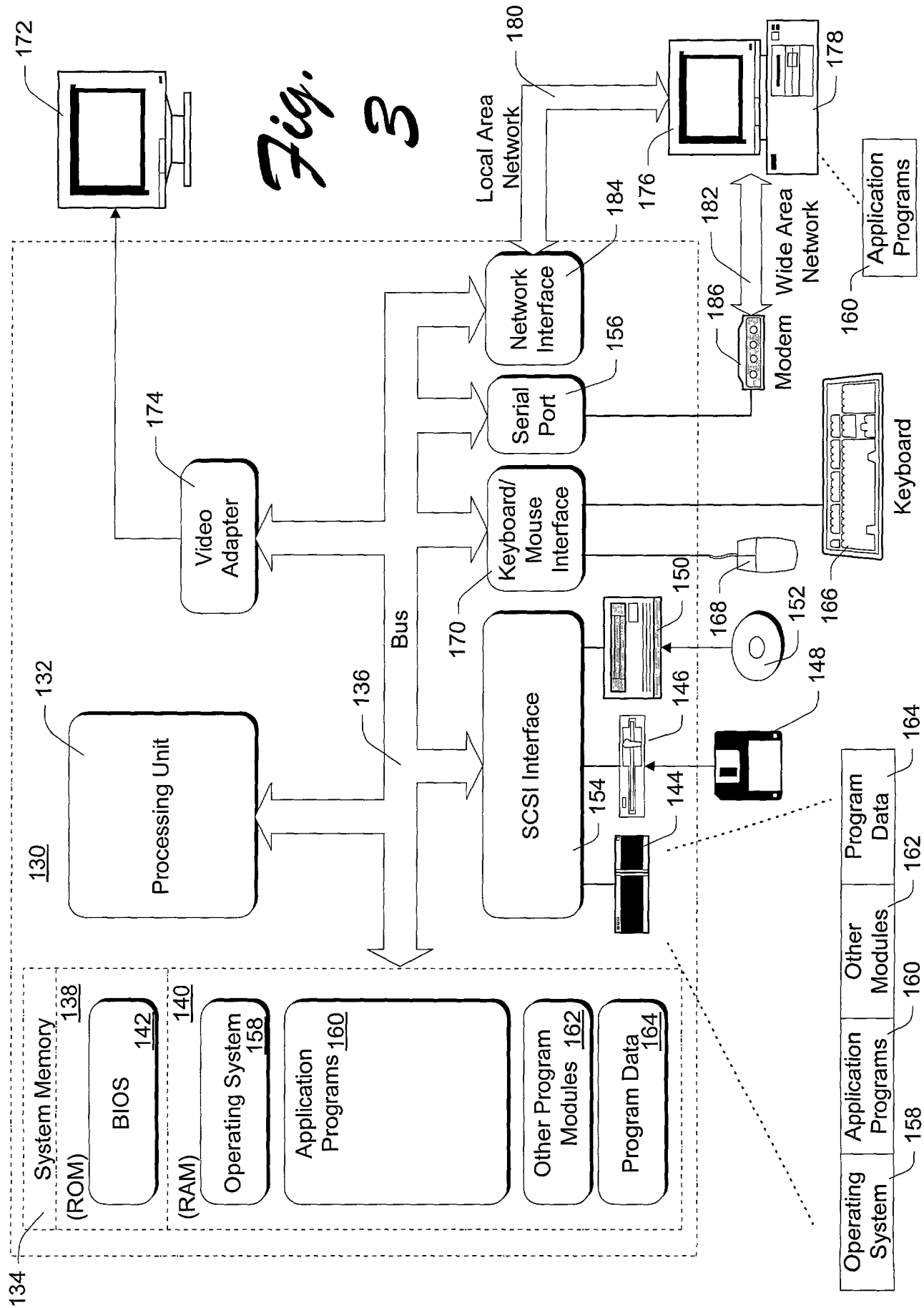


Fig. 2



66700-533330

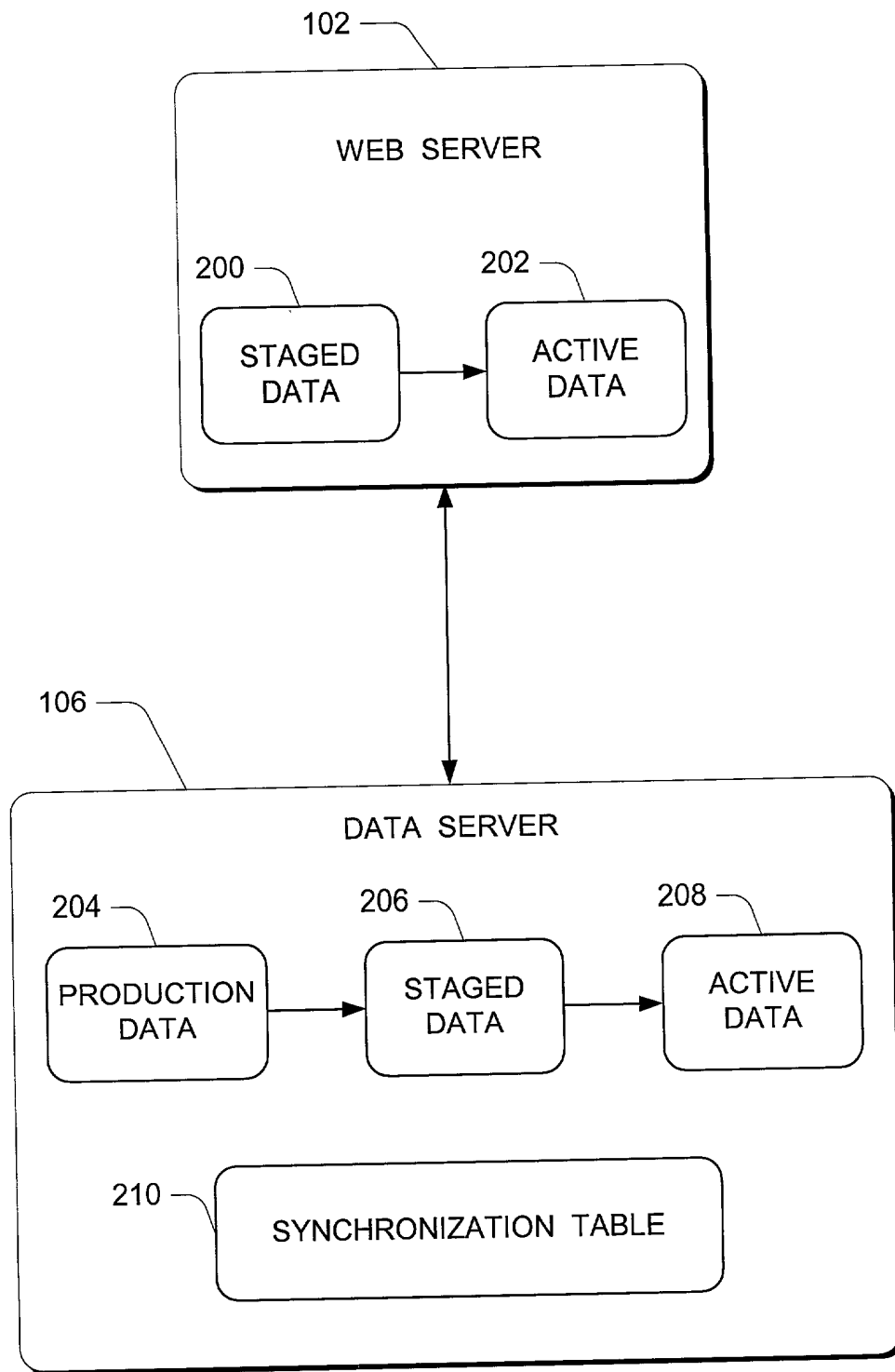
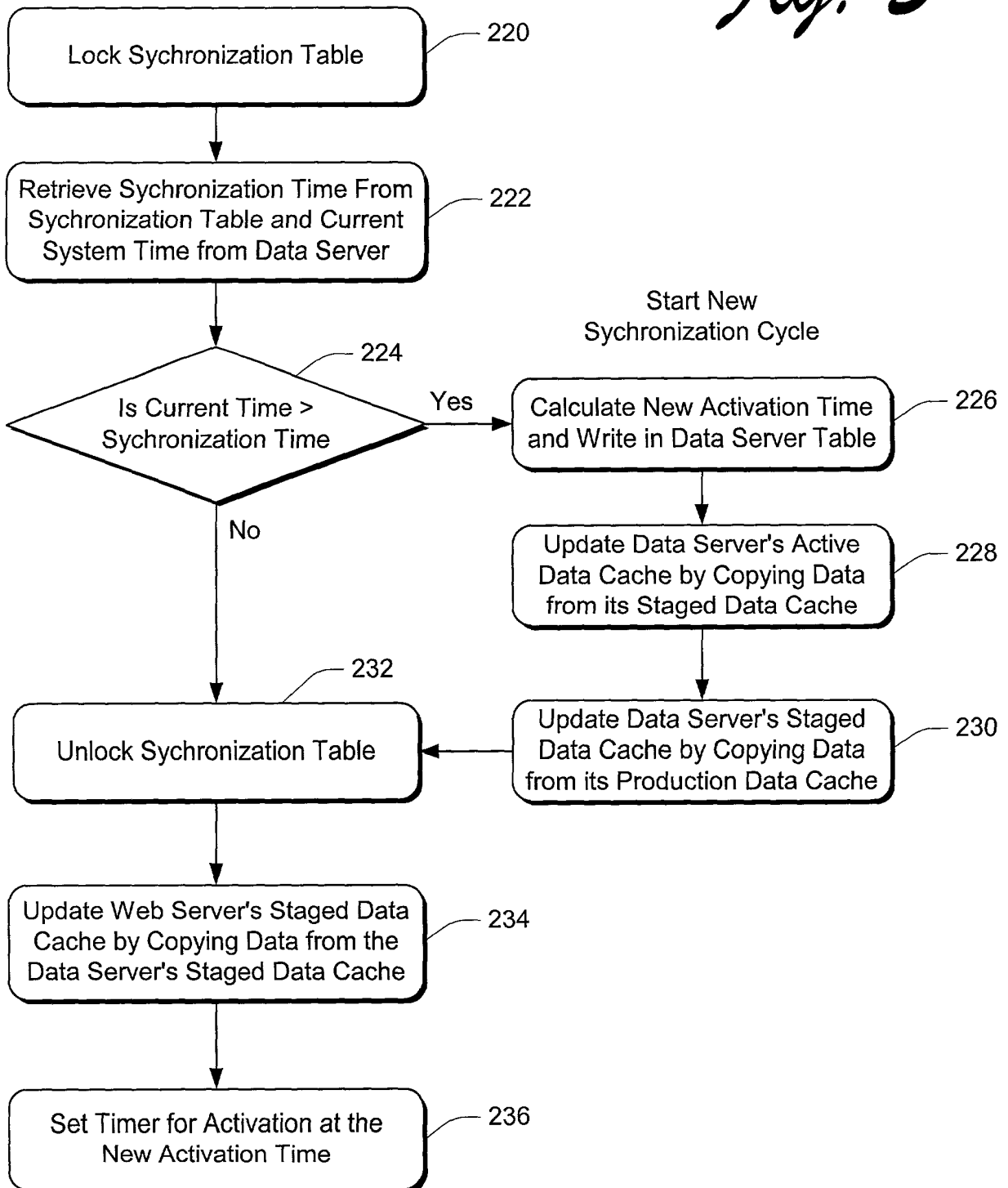


Fig. 4

Fig. 5



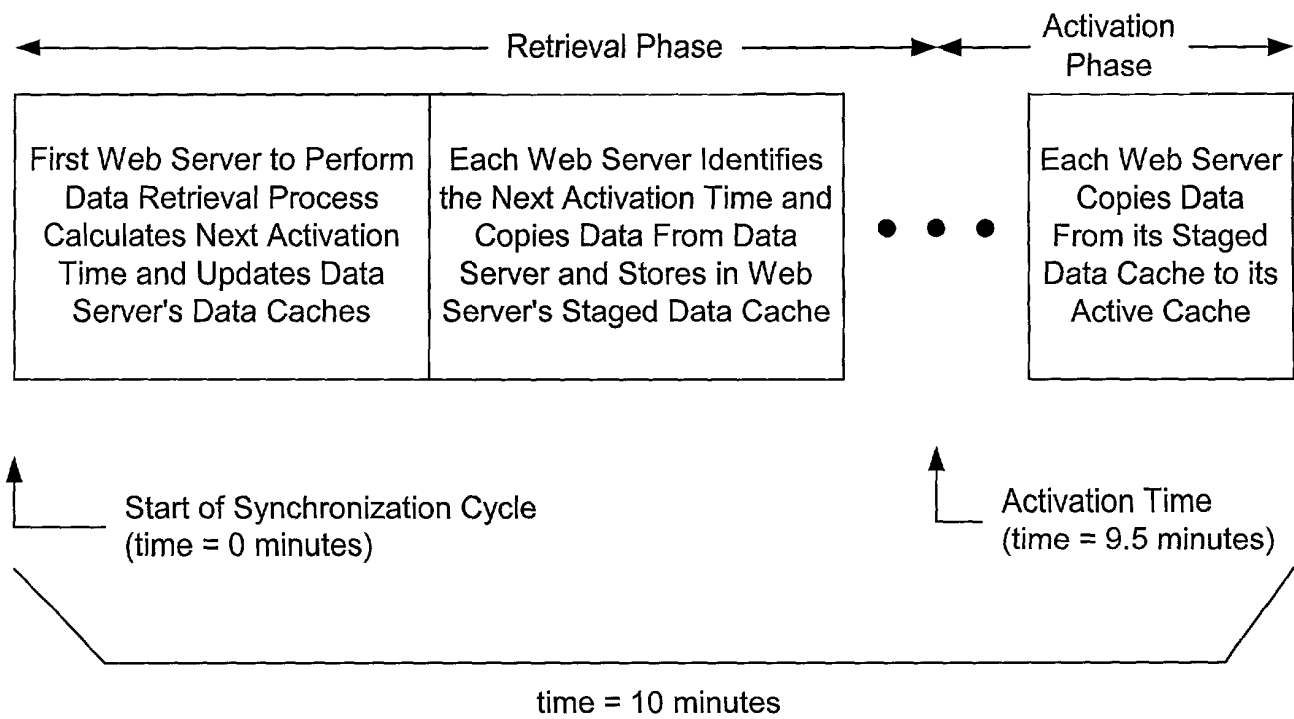


Fig. 7

1 **IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

2 Inventorship Knight et al.
 3 Applicant Microsoft Corporation
 4 Attorney's Docket No. MS1-321US
 5 Title: System and Method for Data Synchronization

6 **DECLARATION FOR PATENT APPLICATION**

7 As a below named inventor, I hereby declare that:

8 My residence, post office address and citizenship are as stated below next to
 9 my name.

10 I believe I am the original, first and sole inventor (if only one name is listed
 11 below) or an original, first and joint inventor (if plural names are listed below) of the
 12 subject matter which is claimed and for which a patent is sought on the invention
 13 entitled "System and Method for Data Synchronization," the specification of which
 14 is attached hereto.

15 I have reviewed and understand the content of the above-identified
 16 specification, including the claims.

17 I acknowledge the duty to disclose information which is material to the
 18 examination of this application in accordance with Title 37, Code of Federal
 19 Regulations, § 1.56(a).

20 PRIOR FOREIGN APPLICATIONS: no applications for foreign patents or
 21 inventor's certificates have been filed prior to the date of execution of this
 22 declaration.

23 **Power of Attorney**

24 I appoint the following attorneys to prosecute this application and transact all
 25 future business in the Patent and Trademark Office connected with this application:
 26 Lewis C. Lee, Reg. No. 34,656; Daniel L. Hayes, Reg. No. 34,618; Allan T.

1 Sponseller, Reg. 38,318; Steven R. Sponseller, Reg. No. 39,384; James R.
2 Banowsky, Reg. No. 37,773; Lance R. Sadler, Reg. No. 38,605; Michael A. Proksch,
3 Reg. No. 43,021; Thomas A. Jolly, Reg. No. 39,241; David A. Morasch, Reg. No.
4 42,905; Katie E. Sako, Reg. No. 32,628 and Daniel D. Crouse, Reg. No. 32,022.

5 Send correspondence to: LEE & HAYES, PLLC, 421 W. Riverside Avenue,
6 Suite 500, Spokane, Washington, 99201. Direct telephone calls to: Steven R.
7 Sponseller (509) 324-9256.

8
9 All statements made herein of my own knowledge are true and that all
10 statements made on information and belief are believed to be true; and further that
11 these statements were made with the knowledge that willful false statements and the
12 like so made are punishable by fine or imprisonment, or both, under Section 1001 of
13 Title 18 of the United States Code and that such willful false statement may
14 jeopardize the validity of the application or any patent issued therefrom.

15
16 * * * * *

17 Full name of inventor: Kenneth J. Knight

18 Inventor's Signature

Kenneth J. Knight Date: 8/31/99

19 Residence: Redmond, WA

20 Citizenship: USA

21 Post Office Address: 21024 NE 36th Street
22 Redmond, WA 98053
23
24
25

Full name of inventor:

David J. Messner

Inventor's Signature

 Date: 8/31/99

Residence:

Redmond, WA

Citizenship:

US

Post Office Address:

6602 149th Avenue NE
Redmond, WA 98053

6602 149th Avenue NE